**PRACTICAL NO:11**

**PROGRAM:**

#include<iostream>

#define MAX 10

using namespace std;

struct queue{

int data[MAX];

int front,rear;

};

class Queue{

struct queue q;

public:

Queue(){q.front=q.rear=-1;

}

int isempty();

int isfull();

void enqueue(int);

int delqueue();

void display();

};

int Queue::isempty(){

return (q.front==q.rear)?1:0;

}

int Queue::isfull(){

return(q.rear==MAX-1)?1:0;

}

void Queue::enqueue(int x){

q.data[++q.rear]=x;

}

int Queue::delqueue(){

return q.data[++q.front];

}

void Queue::display(){

int i;

cout<<endl;

for(i=q.front+1;i<=q.rear;i++){

cout<<q.data[i]<<" ";

}

}

int main(){

Queue obj;

int ch,x;

do{

cout<<"\n1.Insert job\n2.Delete data\n3.Display\n4.Exit\nEnter your choice:";

cin>>ch;

switch(ch){

case 1:

if (!obj.isfull()) {

cout<<"\nEnter data:\n";

cin>>x;

obj.enqueue(x);

cout<<endl;

}

else{

cout<<"Queue is overflow !!!\n\n\n";

}

break;

case 2:

if(!obj.isempty())

{

cout<<"\n Deleted element="<<obj.delqueue()<<endl;

}

else{

cout<<"\n Queue is underflow\n\n";

cout<<"\n Remaining jobs:\n";

obj.display();

}

break;

case 3:

if(!obj.isempty())

{

cout<<"\n Queuw contains:\n\n";

obj.display(); }

else{

cout<<"\n Queue is empty!!!\n\n";

}

break;

case 4:

cout<<"\nExiting program....";

}}while(ch!=4);

return 0;

}

**OUTPUT**:

1.Insert job

2.Delete data

3.Display

4.Exit

Enter your choice:1

Enter data:

2

1.Insert job

2.Delete data

3.Display

4.Exit

Enter your choice:1

Enter data:

6

1.Insert job

2.Delete data

3.Display

4.Exit

Enter your choice:2

Deleted element=2

1.Insert job

2.Delete data

3.Display

4.Exit

Enter your choice:3

Queuw contains:

6

1.Insert job

2.Delete data

3.Display

4.Exit

Enter your choice:4

Exiting program....